Rolling Stone(s)

DiPS CodeJam 24

Prompt

You have discovered an ancient mechanism that involves a grid of stones used to balance an old structure. The grid is represented by a map where spherical stones (denoted by O) and angular stones (denoted by #) are placed. The grid can tilt towards one side (up, down, right or left), and when tilted, results in the spherical stones not obstructed by angular stones in the tilt falling off the grid. This tiltable grid is paired with a receptacle to hold as many stones as possible.
When the grid is tilted, spherical stones roll downwards, but cannot roll over angular stones. Angular stones stay fixed in place.
Your task is to find out which way the grid can be tilted so that the least number of stones may be placed in the receptacle to balance the structure.

Input Format

- The first line of the input contains two space-separated integers m and n, denoting the number of rows and columns in the grid respectively.
- The next m lines contain n space-separated characters each, denoting the map. O indicates a spherical stone, # indicates an angular stone, and . indicates a blank space.

Output Format

The first and only line of your output must contain a single integer i, denoting the least number of stones may be placed in the receptacle to balance the structure.

Constraints

- $10^2 \leq m, n \leq 10^3$

Sample Program

```
# Assuming that the grid has been parsed from input into a 2-dimensional array

def solve(grid):
  stones_on_receptacle = []

  for i in range(4):
    grid = rotate90(grid)
    unbalanced = 0
    for row in grid:
      row_split = [[]]
      for c in row:
        if c == "#":
          row_split.append([])
```

```
      else:
        row_split[-1].append(c)

    for z in row_split[:-1:]:
      unbalanced += z.count("O")
  stones_on_receptacle.append(unbalanced)

return sorted(stones_on_receptacle)[0]
```